Claims

1. A garbage collection system that frees memory areas corresponding to objects that are no longer required in an execution procedure of an object-oriented program composed of a plurality of threads, the garbage collection system comprising:

a selection unit operable to select the threads one at a time;

an examination unit operable to execute examination processing with respect to the selected thread, the examination processing including procedures of stopping execution of the thread, finding an object that is accessible from the thread by referring to an object pointer, managing the found object as a non-freeing target, and resuming execution of the thread;

a detection unit operable to, when having detected, after the selection unit has commenced selecting, that an object pointer has been processed as a processing target by a currently-executed thread, manage an object indicated by the processing target object pointer, as a non-freeing target; and

a freeing unit operable to, after the examination processing has been completed with respect to all of the threads, free memory areas that correspond to objects other than the objects that are managed as non-freeing targets.

2. The garbage collection system of Claim 1, wherein the detection unit performs the detection only when the currently-executed thread has not yet been subject to examination processing, and

the detection unit includes:

41

a finding sub-unit operable to, when having performed the detection, store, to a working memory area that corresponds to the currently-executed thread, the processing target object pointer and an object pointer in an object that can be reached

5   from the processing target object pointer; and

a management sub-unit operable to, while execution of a thread is being stopped by the examination unit, manage, as a non-freeing target, an object that can be reached from the object pointer in the working memory area corresponding

10   to the thread.

3. The garbage collection system of Claim 2, wherein

the examination processing is processing for, when an object indicated by an object pointer in a stack corresponding to the selected

15   thread is found to be accessible, repeatedly performing a procedure of, only when both (a) the accessible object is not already being managed as a non-freeing target and (b) an object pointer exists in the accessible object, further finding that an object indicated by the object pointer in the accessible object is accessible,

20   the selection unit, after a first selection, further performs selection if, after the examination processing has been performed by the examination unit, any threads out of the plurality of threads remain that have not been subject to the examination processing, and

25   the selection unit refers to information about the threads, and makes the selection based one or more predetermined thread selection conditions.

4. The garbage collection system of Claim 3, wherein

the thread selection conditions include a condition indicating that any threads whose thread state is a wait state are to be selected before any threads whose thread state is a state other than the wait state, and

if a thread whose thread state is the wait state exists when making the selection, the selection unit selects the thread whose state is the wait state.

5. The garbage collection system of Claim 4, wherein

the thread selection conditions include a condition indicating that any threads whose thread priority level is low are to be selected before any threads whose thread priority level is high.

6. The garbage collection system of Claim 5, wherein

the thread selection conditions include a condition indicating that any threads whose corresponding stack size is small are to be selected before any threads whose corresponding stack size is large.

7. The garbage collection system of Claim 6, further including a memory management mechanism that manages memory with use of a memory management unit (MMU),

wherein each time an object is to be generated, a memory area corresponding to the object is allocated by the memory management mechanism, and

the freeing unit frees the memory areas via the memory management mechanism.

43

8. The garbage collection system of Claim 3, wherein

the thread selection conditions include a condition indicating that any threads whose corresponding stack size is small are to be selected before any threads whose corresponding stack size is large.

9. The garbage collection system of Claim 3, wherein

the thread selection conditions include a condition indicating that any threads whose thread priority level is low are to be selected before any threads whose thread priority level is high.

10. The garbage collection system of Claim 1, using a memory management mechanism that manages memory with use of a memory management unit (MMU),

wherein each time an object is to be generated, a memory area corresponding to the object is allocated by the memory management mechanism, and

the freeing unit frees the memory areas via the memory management mechanism.

11. A garbage collection method that, in a computer, frees memory areas corresponding to objects that are no longer required in an execution procedure of an object-oriented program composed of a plurality of threads, the garbage collection method comprising:

a selection step of selecting the threads one at a time;

an examination step of executing examination processing with respect to the selected thread, the examination processing including procedures of stopping execution of the thread, finding an object that is accessible from the thread by referring to an object pointer,

44

managing the found object as a non-freeing target, and resuming execution of the thread;

a detection thread of, when having detected, after the selection in the selection step has commenced, that an object pointer has been processed as a processing target by a currently-executed thread, manage an object indicated by the processing target object pointer, as a non-freeing target; and

a freeing step of, after the examination processing has been completed with respect to all of the threads, freeing memory areas that correspond to objects other than the objects that are managed as non-freeing targets.

12. The garbage collection method of Claim 11, wherein

the computer uses a memory management mechanism that manages memory with use of a memory management unit (MMU), and, in an execution procedure of an object-oriented program, each time an object is to be generated, allocates a memory area corresponding to the object according to the memory management mechanism, and

the freeing step frees the memory areas via the memory management mechanism.

13. A computer program for having a computer execute garbage collection processing that frees memory areas corresponding to objects that are no longer required in an execution procedure of an object-oriented program composed of a plurality of threads, the garbage collection processing including:

a selection step of selecting the threads one at a time;

an examination step of executing examination processing with

45

respect to the selected thread, the examination processing including procedures of stopping execution of the thread, finding an object that is accessible from the thread by referring to an object pointer, managing the found object as a non-freeing target, and resuming

5 execution of the thread;

a detection thread of, when having detected, after the selection in the selection step has commenced, that an object pointer has been processed as a processing target by a currently-executed thread, manage an object indicated by the processing target object pointer,

10 as a non-freeing target; and

a freeing step of, after the examination processing has been completed with respect to all of the threads, freeing memory areas that correspond to objects other than the objects that are managed as non-freeing targets.

15

14. The garbage collection method of Claim 13, wherein

the computer uses a memory management mechanism that manages memory with use of a memory management unit (MMU), and, in an execution procedure of an object-oriented program, each time an object is to

20 be generated, allocates a memory area corresponding to the object according to the memory management mechanism, and

the freeing step frees the memory areas via the memory management mechanism.